MILO: requirements

We developed requirements based on internal staff discussions and talking with EDS staff

Why not with students?

1. It should accurately measure indoor air quality **
2. It should be portable ***
3. It should be possible to get the data off the device **
4. It should be a useful pedagogical exercise ***
5. It should maintain privacy *
6. It should be low cost *
7. It should be rugged and robust **
8. Multiple systems should be able to be used simultaneously ***
9. It should be easy to view the current and past data **
10. It should leverage MIT facilities **
What makes a good specification? No single approach for all of HW & SW

- It might be a well-defined metric and value (or range of values)
  - Example: BOM $\leq$ $100$
  - Example: Measurement interval $\leq$ 10 min
- It could be qualitative
  - Example: HTTPS GET/POST for server comms
- It could directly imply a particular implementation
  - Example: Connectivity: WiFi 802.11a/b/g/n [2.4 GHz]
- Or you might not know what it should be yet
  - Example: Sensor accuracy: ???
- Or, you might not even know about that specification
  - Example: ???

A good spec is verifiable...else how do you know if you meet the specs and thus the requirements?
Don’t get hung up if you don’t know many of the specs at the beginning

The two most important points:

1. **Have a plan**: Work hard to plan ahead...and adjust the plan as needed

2. **Write stuff down**: Your team should have a single specifications document – a common understanding
MILO specifications [1/2]

• **Financial**
  - BOM <= $100 for electronics components, PCB
  - BOM: TBD for enclosure, mechanical parts
  - Time to market: ~8 weeks

• **Regulatory**
  - FCC certification for WiFi radio module

• **Industrial design**
  - Weight: < 300 g (~2 iphones)
  - Size: <10 x 10 x 10 cm [kinda small]
  - Survive 12” drop onto table
  - Enclosure materials: 3DP plastics available in EDS, laser-cut plastics available in EDS

• **Environmental**
  - Operating temperature: 0 to 70°C [commercial temp range]
  - Humidity: 10 to 95% RH

• **Engineering**

  • **Sensors**
    - Air quality: TBD
    - T: 0 to 70 °C
    - RH: 10 to 95% RH
    - Measurement interval: <=10 min

  • **Compute**
    - MCU: TBD
    - Firmware in C/C++

  • **Comms**
    - At least WiFi 802.11a/b/g 2.4 GHz
    - 5 GHz would be nice [801.22n]
    - WPA2-Enterprise w/ PEAP (MSCHAPv2) authentication and TLS encryption [this is what MIT Secure wants]

  • **Energy management**
    - LiPo battery
    - Lifetime between charging: >12 h

  • **Server**
    - Machine TBD, one for each student
    - SSH access for students, and staff
    - OS: Linux
    - Web server: NGINX
    - HTTPS GET/POST connections
    - DB: SQLite
MILO specifications [2/2]

- **Firmware**
  ![Block diagram](image)

- **Still to specify**
  - How to reset?
  - Data processing and what is transmitted
  - Sleep state, interval

- **Software [on server]**
  - Store data perpetually in SQLite table
    - Fields: Index number, Timestamp, RH, T, AQ measurements
    - No location information transmitted (or stored)
  - Web front-end
    - Framework: TBD

Here we see that SW requirements often are specified differently [block diagram, wireframe, state machine, text] than HW

Does this cover all the requirements?
Next, we iterate:

- Market research: what’s out there and available, what do our competitors do?
- Draw concepts: form and function
  - This will involve system design and partitioning
- Identify high-risk aspects
- De-risk via:
  - Short-loop prototyping
  - Modeling
  - Research [incl. more market research]
- Update specifications document as needed ← remember this is a working document
- Once you have a system design & partition that is suitably stable – go ahead and start detailed design
  - Knowing this may/will iterate back into specs
MILO: market research

• This is an active space
  • For-profit, non-profit, DIY

---

AirVisual Sensors

AirVisual Series

Everything you need to monitor the air quality inside and outside your home or place of business. The indoor air monitor measures indoor air quality and displays outdoor air quality from the paired outdoor monitor.

<table>
<thead>
<tr>
<th>Replacement Sensors</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>AirVisual Pro</td>
<td>$289.00</td>
</tr>
<tr>
<td>AirVisual Outdoor</td>
<td>$289.00</td>
</tr>
<tr>
<td>AirVisual Bundle</td>
<td>$549.00</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Sensor Specifications</th>
<th>AirVisual Pro</th>
<th>AirVisual Outdoor – 2-PM</th>
</tr>
</thead>
<tbody>
<tr>
<td>PM (Particulate Matter)</td>
<td>0.3 - 2.5 μm</td>
<td>PM2.5: 0.3 - 2.5 μm PM10: 0.3 - 10.0 μm PM1: 0.3 - 1 μm</td>
</tr>
<tr>
<td>CO₂ (Carbon Dioxide)</td>
<td>400 - 10,000 ppm (parts per million)</td>
<td>N/A</td>
</tr>
<tr>
<td>Temperature</td>
<td>14 to 104 °F (-10 to 40 °C)</td>
<td>-22 to 140°F (-30 to 60°C)</td>
</tr>
<tr>
<td>Humidity</td>
<td>0 - 95%</td>
<td>0 - 100% RH, non-condensing</td>
</tr>
</tbody>
</table>

www.iqair.com/us/air-quality-monitors
NEW: Indoor Air Quality Monitor / PurpleAir PA-I-LED

$209.00

The PurpleAir PA-I-LED air quality monitor’s built-in WiFi integration will allow you to check your air quality through the real-time PurpleAir Map – from anywhere in the world. **Double tap to adjust the brightness** of the highly-visible multi-colored LED ring, allowing quick air quality identification from across the room. Uncluttered and attractive, the PurpleAir indoor monitor provides you and your family with industry-leading performance in measuring PM2.5 pollutant levels in your home.
MILO: market research

FLOW 2

199 €
Pollutants: Measures PM1, PM2.5, PM10, NO2, VOCs
Battery: Typical daily use charge of 24-72h depending on use of idle mode
Connection: Bluetooth Low Energy (BLE)
Charging: USB-C & custom metal contacts
Strap: Silicone
Color: Graphite Grey

plumelabs.com/en/flow/
MILO: market research

• Many products out there measure PM10, PM2.5, VOC, NOx (and T, RH)
  • But CO2? Pressure?
• AQI: Comprised of $\text{PM}_{2.5}$, $\text{PM}_{10}$, NO$_2$, SO$_2$, CO, O$_3$ [1h & 8h]

What about these?

IQAir FCC cert teardown

Flume FCC cert teardown

Purpleair teardown

drive.google.com/file/d/11Y-x0m3KHEleb5fRSV8UtnKr9MZzPcdv/view

fccid.io/2AMBQ-N1/Internal-Photos/Internal-Photos-01-3640386

fccid.io/2APMO-FLOW/Internal-Photos/Internal-Photos-3952125
MILO: market research

- What sensors are commercially available?
- COTS: commercial off-the-shelf

**Particulate: PM2.5, PM10**

- **Plantower PMS series [1003, 3003, etc.]**
  - Price: ???
  - $10-20/ea

- **Honeywell HPM series**
  - Price: $70-80/ea

- **Sensirion SPS30**
  - Price: $30-50/ea

- **Plantower PIRS10A**
  - Price: ???
MILO: market research

• T/RH are readily available
  • Lots of specs, sizes, costs, etc.
• VOC/NOx also readily available
• Others are less common and/or expensive
  • O₃: $20-50/ea
  • CO: most are $20-50+/ea
  • SO₂: $20+/ea

Temperature, humidity
Typically bundled together...why?

• TE connectivity TSY02S
  $3-4/ea @10
  ~7 parts in their product line
• Sensirion SHTC3
  $2-3/ea @10
  ~25 parts in their product line

Formaldehyde/VOC/NOx

• Plantower DS-HCHO-20
  ~$25/ea
• Sensirion SGP41
  $5-8/ea @10
• Bosch BME680
  ~$10/ea @10
Next, let’s sketch some concepts & systems

We need to consider

- Industrial design: what it “looks-like”
- Engineering: how it functions

We can “sketch”

- On paper with pen or pencil
- On computer in ppt, solidworks, fusion360, etc.
MILO: concepts & systems

• A first system sketch
• We’ve already decided to forgo O$_3$, SO$_2$
  • Too costly
  • Extra complexity not worth the pedagogical value or “time-to-market”
  • Not included in most consumer products
MILO: system design & partitioning

• Our system block diagram starts to imply a system partition
  • Functional partitioning: allocating functions to different parts of the system
  • Physical partitioning: What parts go where, how do they physically connect to each other

• Partitioning can be applied recursively
  • Big blocks into smaller subblocks

• How far to go?
  • As far as needed to make it clear what to design, and so a person/team can start to design

• We partition to manage complexity
  • Subsystems can be designed independently as long as interface is well-defined
  • Allows abstracting away details of other subsystems
MILO: system design & partitioning

• A good partitioning will have parts that
  • Make internal sense – are coherent in terms of the functionality
    • WiFi + RH/T sensor? Probably not
    • RH + T? Maybe
  • Minimize coupling between parts
    • Minimize interfaces
    • Interfaces often translate to connectors, wires, cables, tubes, APIs, function calls, methods, etc.
    • Strong coupling can suggest that parts belong together rather than separate
  • In a company, partitions may be organized by team for each subsystem
    • Sensors/electronics, power, firmware, mechanical, industrial, SWE, backend, frontend, etc.

• There is no optimal partition...
MILO: system partitioning & tradeoffs

- How do we evaluate/compare designs?
- Trade-off analysis
  - Translate a design back into specs: Performance, cost, size, power, etc.
  - Tradeoff implies that there is no single optimum – it’s up to you as the designer to choose!
- Identify
  - High-risk and addressable unknowns
- De-risk
  - Research
  - Model
  - Prototype
- Once your system diagram is stable…move onto detailed design
MILO specifications [1/2]

• **Financial**
  • BOM <= $100 for electronics components, PCB
  • BOM: TBD for enclosure, mechanical parts
  • Time to market: ~8 weeks

• **Regulatory**
  • FCC certification for WiFi radio module

• **Industrial design**
  • Weight: < 300 g (~2 iPhones)
  • Size: <10 x 10 x 10 cm [kinda small]
  • Survive 12” drop onto table
  • Enclosure materials: 3DP plastics available in EDS, laser-cut plastics available in EDS

• **Environmental**
  • Operating temperature: 0 to 70°C [commercial temp range]
  • Humidity: 10 to 95% RH

• **Engineering**
  • **Sensors**
    • Air quality: NOx/VOC, PM
      • Accuracy: ???
    • T: 0 to 70 °C
      • Accuracy: ???
    • RH: 10 to 95% RH
      • Accuracy: ???
    • Measurement interval: <=10 min
  • **Compute**
    • MCU: TBD
    • Firmware in C/C++
  • **Comms**
    • At least WiFi 802.11a/b/g 2.4 GHz
    • 5 GHz would be nice [801.22n]
    • WPA2-Enterprise w/ PEAP (MSCHAPv2) authentication and TLS encryption [this is what MIT Secure wants]
  • **Energy management**
    • LiPo battery
    • Lifetime between charging: >12 h
  • **Server**
    • Machine TBD, one for each student
    • SSH access for students, and staff
    • OS: Linux
    • Web server: NGINX
    • HTTPS GET/POST connections
    • DB: SQLite

**Updates to spec**

**Things to worry about**

Ver. 2
MILO specifications [2/2]

• Firmware

• Still to specify
  • How to reset?
  • Data processing and what is transmitted
  • Sleep state, interval

• Software [on server]
  • Store data perpetually in SQLite table
    • Fields: Index number, Timestamp, RH, T, AQ measurements
    • No location information transmitted (or stored)
  • Web front-end
    • Framework: TBD

Web wireframe

Ver. 2
MILO: de-risking

• Some high-risk questions
  • Can we implement in ~8 weeks?
    • Other instructors say PCB design can’t work in class setting...
    • So we tested this out a bit over Fall and IAP, and are testing with you!
  • How hard will it be to make these AQ measurements?

• Medium-risk
  • Can we achieve 12 h lifetime?
    • Let’s model
    • Rest of electronics are small and light, so plenty of room for beefy battery if needed

What now?
• Research & discuss with team
  • Sensor parts & specs
  • MCU & WiFi choices
  • PCB fab houses
• And iterate! Update system design, specs, concepts
MILO: from specifications to design

- Much of the content over the next few weeks is intended to help you design systems
  - Labs, psets, and lectures
  - Sensors, compute, comms, etc.
- Let’s look at a few parts in detail
MILO: sensors design

• Sensors
  • Connect to MCU, but partitioned separately
  • Because ~0 MCUs have integrated sensors
    • T is exception...more in a few weeks
  • Some sensors do have integrated MCUs
    • Such as for incorporating processing, AI, etc.
    • Reduce part count on board
    • But typically constrained functionality

• RH/T
  • Together, or partition?
  • Almost all RH sensors also include T, so no benefit to separate T
  • ~all RH/T sensors have digital outputs

• VOC/NOx
  • Typically all-in-one part

• PM
  • Decide to remove b/c of:
    • Cost
    • Reasonable tradeoff of pedagogy vs. utility
MILO: sensors design

• What is the interface between sensors and MCU?
  • Physical interface
    • Chip-level comms is often via I2C, SPI (sometimes UART)
    • 2+ traces on PCB, 2+ pins on MCU
      • More MCU pins ➔ bigger MCU (sometimes), more expensive
  • Functional interface
    • A digital communications protocol: I2C, SPI most common
    • An API/library
      • MCU should have the needed communications peripheral (else you have to bit-bang your own)
      • A set of commands from sensor manufacturer OR a library that encapsulates those commands

The datasheet is your friend
MILO: MCU & comms design

- MCU & comms
  - Details in a few weeks, but...

- Factors influencing choice
  - MCU family
  - MCU w/ or w/o integrated WiFi
  - Peripherals to connect to display, sensors, etc.
  - Price & availability
  - RAM, Flash, etc. SW affects HW choice!
  - Use in other classes
  - Etc..

- Examined a few options
  - ATTiny, STM32 family
  - Teensy 4.0 + WiFi module [such as ESPxx]
    - NXP IMXRT1062DVL6 w/ ARM Cortex-M7
  - ESP32C3 [MCU + WiFi]
    - Espressif RISC-V Core

<table>
<thead>
<tr>
<th></th>
<th>Teensy4.0 + WiFi</th>
<th>ESP32C3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cost</td>
<td>$20+ @ 10</td>
<td>$2.10 @ 10</td>
</tr>
<tr>
<td>Peripherals [I2C, SPI, ADC, UART, USB]</td>
<td>All that are likely needed</td>
<td>Same</td>
</tr>
<tr>
<td>Avail</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>Used in other classes?</td>
<td>6.200, 6.310</td>
<td>6.190</td>
</tr>
<tr>
<td>Number of parts</td>
<td>2 [+passives]</td>
<td>1 [+passives]</td>
</tr>
</tbody>
</table>
MILO: MCU & comms design

• MCU & comms programming
  • Originally wanted plain C/C++

• Question: how hard to get WiFi stack up-and-running on bare ESP32C3?

• Short-loop prototyping
  • Get ESP32 dev board
  • Try it out!

• Answer: pretty hard
  ➔ Incompatible with time-to-market

• Use Arduino libraries as needed [but no Arduino IDE!]
Prototyping for de-risking

- Some HW aspects we can design and de-risk by research and modeling
- Others require prototyping
- Breakouts are fast/easy way to get started
  - With hardware design, firmware design
  - Breakouts are also useful inspiration when it is time to design your own board
- Many/most of these are available substantially cheaper from China
  - Though may take longer to arrive, may be *sketchy*
- There are also evaluation kits
- For other parts (such as SMT ICs), you can get adapter boards

We have parts available, or can order for your team!

- Adafruit Bosch BME680 breakout
  - $19 @ 1
- Adafruit Sensirion SGP40 breakout
  - $15 @ 1
- ESP32-C3 dev board
  - $3.30 @ 1
- SMD adapter boards
MILO: server design

• Server-side architecture
• Server
  • Cloud provider, like AWS, GCP, etc.?
  • Virtual machines at MIT?
  • Physical machines at MIT?
• Database
  • SQLite, MySQL, etc.
• Web framework
  • Django, Flask, Plot.ly, etc.
• For all these components, pedagogical utility was the primary consideration
  • Expose students to inner workings, do not “black box” unless absolutely necessary
  • As simple as possible, yet authentic
  • Easy to manage and help debug
  • Potential to scale for future course offerings

NGINX web server

External client [phone, laptop]

Browser at website

Python GET/POST request handler

SQLite dB

RPi 3 or 4 w/ Raspbian
MILO specifications [1/2]

- **Financial**
  - BOM <= $100 for electronics components, PCB
  - BOM: TBD for enclosure, mechanical parts
  - Time to market: ~8 weeks

- **Regulatory**
  - FCC certification for WiFi radio module [part of ESP32C3 module]

- **Industrial design**
  - Weight: < 300 g [~2 iPhones]
  - Size: <10 x 10 x 10 cm [kinda small]
  - Survive 12” drop onto table
  - Enclosure materials: 3DP plastics available in EDS, laser-cut plastics available in EDS

- **Environmental**
  - Operating temperature: 0 to 70°C [commercial temp range]
  - Humidity: 10 to 95% RH

- **Engineering**
  - **Sensors**
    - Air quality: NOx/VOC SGP41-D-R4
      - Accuracy is not a provided spec!
    - T: 0 to 70 °C SHTC3-TR-10KS
      - Accuracy: +/- 2% RH
    - RH: 10 to 95% RH SHTC3-TR-10KS
      - Accuracy: +/- 0.2 °C
    - Measurement interval: <=10 min
      - <10 sec response time for SGP41
    - Communications: IIC
  - **Compute**
    - MCU: ESP32-C3-WROOM-02
    - Firmware in C/C++ w/ Arduino libraries as needed
  - **Comms**
    - At least WiFi 802.11a/b/g/n 2.4 GHz
    - WPA2-Enterprise w/ PEAP (MSCHAPv2) authentication and TLS encryption [this is what MIT Secure wants]
  - **Energy management**
    - LiPo battery
    - Lifetime between charging: >12 h
  - **Server**
    - RPi 3 or 4, one for each student
    - SSH access for students, and staff
    - OS: Raspbian
    - Web server: NGINX
    - HTTPS GET/POST connections
    - DB: SQLite

Changes to spec
Things to worry about
MILO specifications [2/2]

**Firmware**

- Still to specify
  - How to reset?
  - Functions/APIs for sensor, display, WiFi
  - Data processing and what is transmitted
  - Sleep state, interval

**Software [on server]**

- Store data perpetually in SQLite table
  - Fields: Index number, Timestamp, RH, T, AQ measurements
  - No location information transmitted (or stored)

**Web front-end**

- Framework: TBD

---

Web wireframe

- Chart showing Air quality over time with line for each sensor.
MILO: Power management – modeling

- Power
  - More in a few weeks...
- Can we estimate the lifetime?
- What’s the biggest energy consumer – usually MCU or comms
  - In our case, WiFi
  - Let’s check ESP32-C3 datasheet
- What about battery?
  - ~infinite number of choices
  - Most common rechargeable choice these days is LiPo
  - Let’s look at 18650 b/c it is used in 6.08
    - 3.7V nominal for single cell
    - Typical capacities ~2500 – 3600 mAh

Assume 400 mA @ 3.3V consumption
Assume 3600 mAh capacity @ 3.3V [assume no energy savings for 3.7V to 3.3V conversion]

~9h if transmitting WiFi continuously...which we aren’t going to be doing ➔ should be ok!
MILO: comms and display

• Many ways to view sensor data
  • Sensor node ➔ on-board display
    • Could remove WiFi/comms entirely in some cases
    • Display adds cost to node
  • Sensor node ➔ Phone ➔ view on app
    • Directly connect MILO node to phone, such as via Bluetooth
    • Avoid cloud server infrastructure – phone is ubiquitous
  • Sensor node ➔ Phone/gateway ➔ server ➔ view on website
    • Use phone or other device to hand off data from sensor node to internet
    • Allows remote data retrieval, data fusion from multiple nodes/users
    • Need to maintain server, write apps for node, phone, and server
  • Sensor node ➔ server ➔ view on web
    • Not very common (due to power consumption)
    • Avoids need for app on phone, which is why we’ll use it!
MILO: specifications

**Financial**
- BOM <= $100 for electronics components, PCB
- BOM: TBD for enclosure, mechanical parts
- Time to market: ~8 weeks

**Regulatory**
- FCC certification for WiFi radio module [part of ESP32C3 module]

**Industrial design**
- Weight: < 300 g [~2 iPhones]
- Size: <10 x 10 x 10 cm [kinda small]
- Survive 12” drop onto table
- Enclosure materials: 3DP plastics available in EDS, laser-cut plastics available in EDS

**Environmental**
- Operating temperature: 0 to 70°C [commercial temp range]
- Humidity: 10 to 95% RH

**Engineering**

- **Sensors**
  - Air quality: NOx/VOC SGP41-D-R4
    - Accuracy is not a provided spec!
  - T: 0 to 70 °C SHTC3-TR-10KS
    - Accuracy: +/- 2% RH
  - RH: 10 to 95% RH SHTC3-TR-10KS
    - Accuracy: +/- 0.2 °C
  - Measurement interval: <=10 min
    - <10 sec response time for SGP41
  - Communications: I2C
- **Compute**
  - MCU: ESP32-C3-WROOM-02
  - Firmware in C/C++ w/ Arduino libraries as needed
  - How to program?
- **Comms**
  - At least WiFi 802.11a/b/g/n 2.4 GHz
  - Hardcoded connection to EECS_Labs [open network]
- **Energy management**
  - LiPo battery – 18650 w/ JST-PH2 2-pin connectors
  - Lifetime between charging: >12 h
  - System voltage: 3.3 V
  - Charge from USB-micro or USB-C connection
- **Server**
  - RPi 3 or 4, one for each student
  - SSH access for students, and staff
  - OS: Ubuntu, any recent LTS
  - Web server: NGINX
  - HTTPS GET/POST connections
  - DB: SQLite

**Changes to spec**

**Things to worry about**

**Ver. 4**
MILO: specifications

• Firmware

  - Initialize sensors, wifi
  - Read sensors
  - Compute VOC & NOx index
  - Append data to buffer

  Wake
  • Sleep ~1 s

  Connect to wifi
  • fail
  • Send data
  • fail
  • Clear buffer

  success

  success

No more display!

• Still to specify
  - How to reset?
  - Functions/APIs for sensor, display, WiFi
  - Data processing and what is transmitted
  - Sleep state, interval

• Software [on server]
  - Store data perpetually in SQLite table
    - Fields: Index number, Timestamp, RH, T, AQ measurements
    - No location information transmitted (or stored)

• Web front-end
  - Framework: Plot.ly

Web wireframe
MILO: updated system diagram

Sensor

- RH/T
- VOC/NOx

ESP32-C3

MCU

WiFi

NGINX web server

Python GET/POST request handler

SQLite dB

Server: RPi 3 or 4 w/ Raspbian

Browser at website

External client [phone, laptop]

Power management IC

battery

USB

Power management
MILO: from requirements to specifications

• Industrial design

• Still to specify
  • Status LEDs?
  • External buttons?
  • How to program?
  • What do connectors and cables look like, where are they situated?
  • How exactly to expose sensor to world

External view

Internal view

3 subsystems ➔ 3 PCBs
Test and verification

• Once you make it, does it work? **Does it meet spec?**
  • How will you debug if/when it doesn’t work?

• Better yet, can you anticipate testing during design?
  • This is so-called design for test
  • Electronics
    • Make sure all signals are easy to probe: test points!!
    • Status LEDs to quickly see what’s going on
    • Jumpers or 0–ohm resistors to connect/disconnect subcircuits
    • Ideally, every pin on a device should be accessible
      • Sometimes you can solve a HW problem with code, scalpel, and a jumper wire
  • Design reviews! Electronics, FW, SW, etc.
    • Find problems early ➔ save $$ and time
  • Test fixtures
    • Example: System with radio comms – how do you do environmental test in a metal chamber?
Test and verification

- Financial
  - BOM <= $100 for electronics components, PCB
  - BOM: TBD for enclosure, mechanical parts
  - Time to market: ~8 weeks

- Regulatory
  - FCC certification for WiFi radio module [part of ESP32C3 module]

- Industrial design
  - Weight: < 300 g (~2 iPhones)
  - Size: <10 x 10 x 10 cm [kinda small]
  - Survive 12” drop onto table
  - Enclosure materials: 3DP plastics available in EDS, laser-cut plastics available in EDS

- Environmental
  - Operating temperature: 0 to 70°C [commercial temp range]
  - Humidity: 10 to 95% RH

And so on...
Test and verification

• FW & SW
  • Tests of each function (unit tests) and overall FW
  • Ideally, not just when everything works, but consider common failures
    • WiFi down...
    • Reset
    • Sensor board disconnected
    • And so on...